MATHEMATICS OF COMPUTATION
VOLUME 38, NUMBER 157
JANUARY 1982

On Estimating the Largest Eigenvalue With the Lanczos Algorithm*

By B. N. Parlett, H. Simon and L. M. Stringer

Abstract. The Lanczos algorithm applied to a positive definite matrix produces good approximations to the eigenvalues at the extreme ends of the spectrum after a few iterations. In this note we utilize this behavior and develop a simple algorithm which computes the largest eigenvalue. The algorithm is especially economical if the order of the matrix is large and the accuracy requirements are low. The phenomenon of misconvergence is discussed. Some simple extensions of the algorithm are also indicated. Finally, some numerical examples and a comparison with the power method are given.

1. Introduction. Let A be a positive definite matrix of order n with eigenvalues

$$0 < \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n.$$

For some applications a rough approximation to λ_n, i.e., the spectral norm of A, is all that is wanted, for others a rough approximation to λ_n/λ_1, i.e., the condition number of A. At the other extreme are nuclear engineers, who often want a good approximation to λ_n, 6 significant decimals at least and the eigenvector as well.

In [1] O'Leary, Stewart, and Vandergraft consider the power method and the associated sequence of Rayleigh quotients $\rho_1, \rho_2, \rho_3, \ldots$. They point out that the often miserable asymptotic convergence rate of the ρ_k is irrelevant unless it is necessary to have

$$\frac{\lambda_n - \rho_k}{\lambda_n - \lambda_{n-1}} < 10^{-2} \quad \text{(say)}.$$

They show that even for the nastiest distribution of λ_i for a 1000 by 1000 matrix we can expect to have ρ_k with one correct decimal after 21 steps.

These observations are interesting, but it does not follow that the power method is the appropriate algorithm for rough approximations to λ_n. We should add that [1] does not claim that the power method is the preferred algorithm although an eager reader might well draw that conclusion.

Here are some points we wish to make.

1. If the user is only interested in the rough order of magnitude of λ_n, then the appropriate action is to compute $\max_i a_{ii}$ and $\|A\|_1$, the maximum column sum. This yields a lower bound and an upper bound with no multiplications. If A is large and sparse, then $\|A\|_1/\lambda_n$ cannot get close to its upper bound $\sqrt{n}$. A better estimate of the ratio is $\sqrt{m}$, where m is the average number of nonzero elements per row.

Received November 11, 1980.
1980 *Mathematics Subject Classification.* Primary 65F15.
* The authors gratefully acknowledge support from ONR contract N00014-69-0200-1017.

0025-5718/82/0000-0476/$04.25

If it is not convenient to access the elements of A, then the techniques described below should be considered. If something better than an order of magnitude estimate of λ_n is wanted, then these techniques should definitely be considered. For interactive or hand held computation the power method with Rayleigh quotients, as in [1], may be preferable to our Lanczos algorithm when only one digit in $\|A\|$ is wanted.

2. The choice of tolerance on the error, which is often delegated to the user, is not a trivial matter. If the user says he wants only one correct decimal and $\lambda_n = 10^3$, then he must be content with any answer exceeding 950. Are all such high eigenvalues indistinguishable for his purposes? He may, on reflection, wish to change his tolerance.

Sometimes users do want an approximation μ which is closer to λ_n than to λ_{n-1}, but not by much, say $(\lambda_n - \mu)/(\mu - \lambda_{n-1}) < 1/10$, unless λ_{n-1} is very close to λ_n, say $(\lambda_n - \lambda_{n-1})/\lambda_n < 10^{-6}$. This is, of course, a more difficult specification to meet. These points are pursued further in Section 6. In our opinion the required accuracy should increase linearly with n.

3. In [1] the authors focussed on the eigenvalue distribution which causes the slowest convergence of the Rayleigh quotients ρ_k. In practice, as those who have used the power method are painfully aware, the most troublesome distributions are quite different. A difficulty which afflicts both the power method and the Lanczos algorithm is *misconvergence*. Consider the following values: $\lambda_n = 1000$, $\lambda_{n-1} = 985$, $\lambda_{n-2} = 983$, $\lambda_{n-3} = 981$, $\lambda_{n-4} = 955$, ... It happens, not infrequently, that the ρ_k converge quite nicely to 985 and settle down there for several steps. After a while the ρ_k will start to increase again noticeably and soon converge to the correct value of 1000.

An impatient criterion for termination will mistake the pause at 985 for convergence to that value. On the other hand a cautious algorithm will be inefficient in timing comparisons. This topic is pursued further in Sections 2 and 6. It would be interesting to quantify the trade-off. As so often occurs in numerical analysis, the real difficulty is the criterion for stopping.

4. The authors in [1] mention that the Lanczos algorithm is more powerful than the power method but suggest that to invoke it to obtain a one-decimal approximation to λ_n is overkill.

It is true that Lanczos codes are usually designed to find several eigenvalue/vector pairs, and the reliable ones are rather cumbersome. However, because of its power the Lanczos algorithm

(i) uses almost the minimum number of matrix-vector multiplications, whatever the required accuracy;

(ii) can cope with misconvergence much more adroitly than can the power method.

For this special problem several features of general Lanczos codes can be discarded, and the stripped down version is quite short; see (2.4). Both Lanczos and the power method require about the same amount of working storage: 2 n-vectors plus some extra cells. Last, but not least, the Lanczos code is well suited to either high or low accuracy calculations.

A separate short program can use the output of the main one to compute the associated eigenvector when that is required.

The aim of this paper is to present our algorithm and to show how it and the power method perform on a variety of eigenvalue distributions. The phenomenon of misconvergence is explored in the process. A simple modification of our program yields increasingly good approximations to λ_n/λ_1.

2. The Lanczos Process. The simple Lanczos algorithm for a symmetric $n \times n$ matrix A computes a sequence of Lanczos vectors $v_1, v_2, v_3, \ldots$ as follows:

(2.1)
1: choose an arbitrary v_1, $\|v_1\| = 1$
2: $u_1 = Av_1$
3: for $j = 1, 2, \ldots$ do
$\alpha_j = u_j^T v_j$
$r_j = u_j - \alpha_j v_j$
$\beta_j = \|r_j\|$
$v_j = r_j/\beta_j$
$u_{j+1} = Av_{j+1} - \beta_j v_j$.

One pass through step 3 is a Lanczos step. These equations can be condensed in matrix form as

$$AV_j - V_j T_j = \beta_j v_{j+1} e_j^T,$$

where $V_j = (v_1, v_2, \quad , v_j)$, $e_j^T = (0, 0, 0, \ldots, 1)$ and

$$T_j = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & & \\ \beta_1 & \alpha_2 & \beta_2 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \beta_{j-2} & \alpha_{j-1} & \beta_{j-1} \\ \cdot & & 0 & \beta_{j-1} & \alpha_j \end{bmatrix}$$

The algorithm terminates if $\beta_j = 0$, and this will happen for some $j \leq n$ in exact arithmetic. The eigenvalues of the tridiagonal matrix T_j, also called the Ritz values, are the Rayleigh-Ritz approximations to eigenvalues of A from the subspace spanned by the vectors $v_1, v_2, \ldots, v_j$. More details on the Lanczos process can be found in [2].

Let ϑ_j be the largest eigenvalue of T_j.

Usually the extreme Ritz values are, even for $j \approx 2\sqrt{n}$, good approximations to the corresponding eigenvalues of A. We therefore propose the following strategy for finding λ_n:

(2.4)
for $j = 1, 2, 3, \ldots$ do
1.1: take a Lanczos step
1.2: compute a narrow interval which contains ϑ_j
1.3: compute a bound on $|\vartheta_j - \lambda|$
1.4: if the bound is small enough then stop.

The details of 1.2, 1.3, and 1.4 will be discussed in the following sections.

Our FORTRAN program for algorithm (2.4) consists of 55 executable statements. This number does not include separate subroutines for computing 1.1, 1.3, and $\delta_k(x)$. None of these exceeds 14 statements.

As Section 6 reveals, we have no guarantee that the eigenvalue λ to which ϑ_i converges is λ_n. However, our convergence criterion will not be met until the (uncomputed) Ritz vector belonging to ϑ_j is a reasonable approximation to λ's eigenvector. This level of caution seems to prevent misconvergence in practice at a moderate cost. The yet more cautious criterion described in Section 6 prolongs the algorithm by 50–100%. The old-fashioned criterion of stopping when the change in the ϑ_j's is less than the tolerance is just not reliable enough as our examples show.

The storage requirements of our algorithm are quite modest because one Lanczos step can be implemented in a way that requires only storage for the n-vectors u and v and $2j$ storage for the α_j and β_j. But as $j \ll n$ (as typical example take $n = 500$ and $j < 50$), the main storage requirement lies in the $2n$ which is comparable to the power method.

We should also remark that if A is sparse (say 10 nonzero elements per row) and of narrow bandwidth w, then each step will require $15n$ operations, and so, even if our algorithm took n steps, the total cost compares favorably with the $2wn^2$ operations required by techniques based on similarity transformations. Comparisons of this algorithm and the accelerated power method are given in Section 6.

3. Computing the Largest Eigenvalue of T_j. Computing the largest eigenvalue of a tridiagonal matrix is neither a new nor a difficult task, and we could have used one of the standard routines from EISPACK or any other similar package. As $\|T_j\|_\infty$ is easily computed, a bisection algorithm on the interval $(\vartheta_{j-1}, \|T_j\|_\infty)$ would have been the obvious choice.

In the special situation which we are considering here there is extra information on hand at each Lanczos step. We have tried to develop an elegant procedure which takes full advantage of the situation and converges quickly. A related algorithm based on the simple recursion formula (3.1) is discussed in [3].

Let LDL^T be the triangular factorization of $T_j - xI$. It turns out that $\delta_j(x) = D_{jj}$, called the bottom pivot, is computed from the simple recurrence:

$$\begin{aligned} &\delta_1(x) = \alpha_1 - x, \qquad x \neq \alpha_1, \\ &\left.\begin{aligned} &\delta_k(x) = \alpha_k - x - \beta_{k-1}^2/\delta_{k-1}(x) \\ &\text{if } \delta_k(x) = 0 \text{ then change } x \text{ slightly and start again} \end{aligned}\right\}, \quad k = 2, 3, \ldots, j. \end{aligned} \tag{3.1}$$

The function $\delta_j(x)$ is a $(j, j-1)$ rational function, whose j zeros are the eigenvalues of T_j and whose $j-1$ poles are the eigenvalues of T_{j-1}. Between its poles $\delta_j(x)$ is monotonically decreasing with a slope less than -1 provided (3.2) holds.

From the previous step we have a satisfactory approximation to $\pi = \vartheta_{j-1}$, the largest pole of $\delta_j(x)$. Our algorithm is based on the following notion: On the interval $(\vartheta_{j-1}, \infty)$ the function $\delta_j(x)$ can be adequately approximated by (2.1) rationals of the form

$$\frac{(\zeta - x)(\mu - x)}{(\pi - x)} \qquad (\mu < \pi < \zeta), \tag{3.2}$$

where ζ and μ are free parameters. Since we know π, the pole, accurately we are simply using a quadratic for the function $(\pi - x)\delta_j(x)$. There is more discussion of this model in [3], and we content ourselves with a picture. The model also has slope < -1 (provided (3.2) holds).

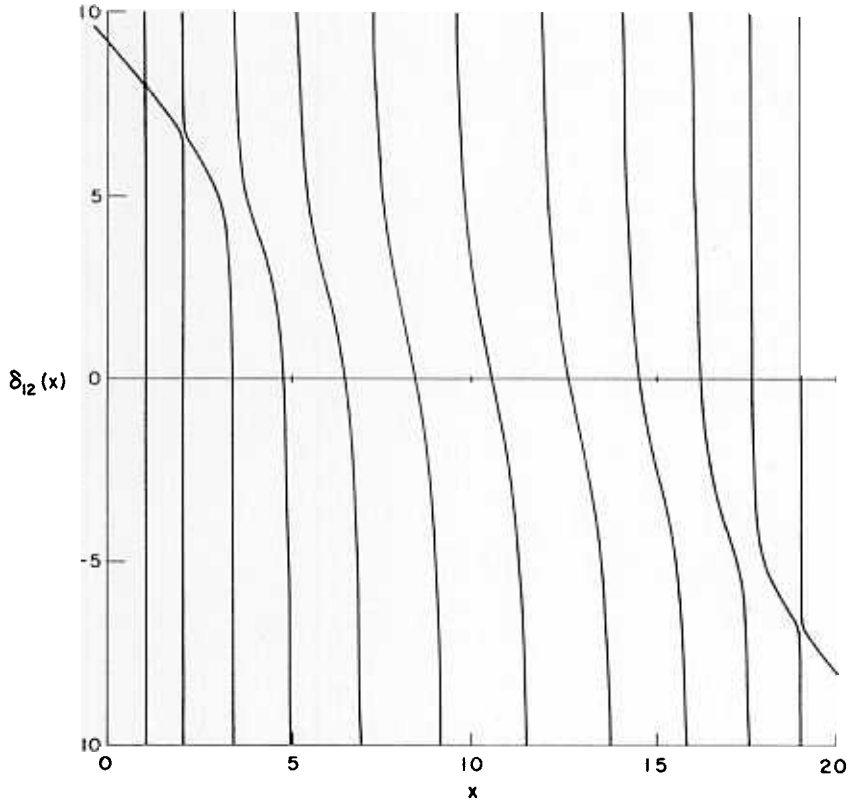

FIGURE 1

The function $\delta_{12}(x)$ for a typical example

We use the model in an iterative scheme based on the algorithm zeroin of Dekker. Given an interval (x_l, x_u), with $\vartheta_j \in (x_l, x_u)$, and a pair of distinct points $x_{k-1}, x_k \in (x_l, x_u)$ along with $\delta_j(x_i)$, $i = k - 1, k$, we interpolate $\delta_j(x)$ by (3.2) at x_{k-1} and x_k in order to find ζ and μ ($\mu < \zeta$). Then we set

$$x_{k+1} = \begin{cases} \zeta & \text{if } \zeta \in (x_l + \text{tol}/2,\ x_u - \text{tol}/2), \\ (x_u + x_l)/2 & \text{otherwise,} \end{cases} \tag{3.3}$$

and then

$$\begin{aligned} x_u &\leftarrow x_{k+1} \quad \text{if } \delta_j(x_{k+1}) < 0, \\ x_l &\leftarrow x_{k+1} \quad \text{if } \delta_j(x_{k+1}) > 0. \end{aligned} \tag{3.4}$$

We keep iterating until $x_u - x_l \leqslant \text{tol}$. The last value of ζ is used as π at the next step. The number of iterations at each step depends on $\text{tol} = \rho\vartheta_j$, where ρ is the required relative accuracy. For $\rho = 10^{-1}$ ($\rho = 10^{-6}$), the average number of iterations is approximately 1.5 (4).

It should be mentioned that in certain cases this iteration need not be invoked at all. Suppose we had stopped the iteration at the $j-1$st Lanczos step with the interval $[x_l, x_u]$. Then, obviously, if after updating $d_u \equiv \delta_j(x_u)$ we still have $d_u < 0$, then this interval also contains ϑ_j, and no further computations are necessary at the jth Lanczos step. If this is the case, we call such an interval stagnant. A stagnant interval does not indicate convergence.

4. Computing a Bound on $|\vartheta_j - \lambda|$. Bounds on the Rayleigh-Ritz approximations to the eigenvalues of A from a subspace are discussed in [2, Chapter 11]. As we do not have any information on the gaps in the spectrum of A, we have to content ourselves with the simple bound

$$|\vartheta_j - \lambda| < \frac{\|Ay - y\vartheta_j\|}{\|y\|}. \tag{4.1}$$

There is at least one eigenvalue λ of A, which satisfies (4.1). Here $y = V_j s$, and s is a normalized eigenvector of T_j corresponding to ϑ_j. Fortunately we do not have to compute y explicitly, because using $y = V_j s$, $T_j s = \vartheta_j s$, and (2.2), the bound (4.1) becomes

$$|\vartheta_j - \lambda| < \frac{\beta_j |e_j^T s|}{\|y\|}. \tag{4.2}$$

We may assume that $\|y\| > 0.9$ (see [2] for justification). As we know β_j from the Lanczos process, we only have to compute $\sigma_j \equiv e_j^T s$, the bottom element of the normalized eigenvector s. This can be done by setting $\xi_j = 1$ and then solving $(T_j - \vartheta_j)x = 0$ for $x = (\xi_1, \xi_2, \ldots, \xi_j)^T$ from bottom to top, except that the top equation is not satisfied: instead we have $(\alpha_1 - \vartheta_j)\xi_1 + \beta_2\xi_2 = \mu$. There is no need to store the ξ_i; instead we accumulate their squares in τ and set $\sigma_j = 1/\sqrt{\tau}$. Moreover, the residual norm $|\mu|/\sqrt{\tau}$ is available as a check; it should be small. The vector x will be a satisfactory approximation to s provided that the starting vector for Lanczos is not almost orthogonal to λ's eigenvector.

The bound $1.1\beta_j\sigma_j$ can be used in a twofold way: it enables us to monitor the convergence of the Ritz values ϑ_j, and it provides a useful starting point for the iteration in Section 3.

5. Starting the Iteration for ϑ_j. Suppose that the interval $[x_l^{(j-1)}, x_u^{(j-1)}]$ was not stagnant at the $j-1$st Lanczos step, so we have to find a new interval and two starting points for the iteration. (An upper index (j) refers here to quantities computed at the jth Lanczos step.) But if $[x_l^{(j-1)}, x_u^{(j-1)}]$ is not stagnant, then we have $\delta_j(x_u^{(j-1)}) > 0$. Hence we can set

$$x_l^{(j)} = x_u^{(j-1)}. \tag{5.1}$$

It is tempting to use $x_l^{(j)}$ in starting the iteration since its δ-value is available. However, when high accuracy is required, $x_l^{(j)}$ is occasionally too close to the pole to be useful. We can get a first starting point using the bound from Section 4:

$$x_1^{(j)} = \pi + \beta_{j-1}\sigma_{j-1}. \tag{5.2}$$

As the quantity $\beta_{j-1}\sigma_{j-1}$ indicates roughly how far we are away from some λ at the $j-1$st Lanczos step, (5.2) is a natural choice.

Then we compute $x_2^{(j)}$ by

$$x_2^{(j)} = \left(x_1^{(j)} + x_l^{(j)}\right)/2. \tag{5.3}$$

Although we could have used something more sophisticated than bisection (e.g., interpolating at x_1 and ∞), our computational experience has shown that this additional work does not reduce the number of iterations significantly. Therefore we chose the simple formula (5.3).

Finally we took as an upper bound

$$x_u^{(j)} = \max\left\{x_l^{(j)}, \alpha_j\right\} + \beta_{j-1}. \tag{5.4}$$

This was motivated by the fact that only in exact arithmetic is $x_1^{(j)}$ an upper bound for ϑ_j, whereas (5.4) is always safe.

At the second Lanczos step, when no previous values for x_u, x_l are available, we compute ϑ_2 directly as the largest eigenvalue of T_2 and then set $x_u^{(2)} = \vartheta_2 + \text{tol}$, $x_l^{(2)} = \vartheta_2 - \text{tol}$. In this way we can begin our new iteration at the third Lanczos step in the manner described above.

6. Misconvergence. Both the power method and the Lanczos algorithm sample the effect of the matrix A at a limited number of vectors, and both can suffer from misconvergence. The simplest, but extreme, case is when the starting vector is orthogonal to the dominant eigenvector and (as happens in exact arithmetic) to all subsequent vectors as well. Then our algorithm will quietly deliver λ_{n-1} instead of λ_n without warning of failure.

In practice, if the starting vector is somewhat orthogonal to the dominant eigenvector, and the separation of λ_n and λ_{n-1} is poor, then it can take many iterations before any Ritz value ϑ_i exceeds λ_{n-1}. In such cases the ϑ_i may well settle down very close to λ_{n-1} for perhaps 10 (or even 100) consecutive iterations before some ϑ_i exceeds $\vartheta_{i-1} + \text{tol}$. Such situations are not at all pathological, as our examples show.

Some quantitative insight into the duration of these false stagnations can be gleaned from an error bound which can be found in [2, Chapter 12, p. 247]. Let ψ_i be the angle between the starting vector and λ_i's eigenvector, and let C_j denote the jth Chebyshev polynomial. Then

$$\left[\frac{\tan \psi_n}{\gamma_n C_{j-1}\left(1 + 2\dfrac{\lambda_n - \lambda_{n-2}}{\lambda_{n-2} - \lambda_1}\right)}\right]^2 \tag{6.1}$$

where

$$\gamma_i \equiv \frac{\lambda_i - \lambda_{i-1}}{\lambda_{i-1} - \lambda_1} \tag{6.2}$$

We are not interested in how long it takes to compute λ_n but in m, the smallest value of j such that $\vartheta_j > \lambda_{n-1}(1 + \rho)$, where ρ is the relative accuracy required. To be specific we estimate m by taking equality in (6.1), and, after making some slight

simplifications, we get

$$C_{m-1}(1 + 2\gamma_n + 2\gamma_{n-1}) = \gamma_n^{-3/2}\tan\psi_n. \tag{6.3}$$

Next we look for k, the first value of j for which $|\vartheta_j - \lambda_{n-1}| < \rho\lambda_{n-1}$. This value may be approximated by ignoring λ_n in the standard error estimate to get

$$\frac{\lambda_{n-1} - \vartheta_j}{\lambda_{n-1} - \lambda_1} \leqslant \left[\frac{\tan\psi_{n-1}}{C_j(1 + 2\gamma_{n-1})}\right]^2 \tag{6.4}$$

We compute k by taking equality in (6.4),

$$C_k(1 + 2\gamma_{n-1}) = \rho^{-1/2}\tan\psi_{n-1}. \tag{6.5}$$

The fundamental assumption underlying this section is that $\cos\psi_n \ll \cos\psi_{n-1}$ ($= 1/\sqrt{n}$, its expected value). The stagnation time which must be endured is $m - k$.

There are still too many unknowns in (6.3) and (6.5), so let us consider a difficult but not unreasonable case, namely $\gamma_n = 2\rho$, $\gamma_{n-1} = 8\rho$. (For example, $\rho = 0.05$, $\lambda_n = 1.0$, $\lambda_{n-1} \approx 0.9$, $\lambda_{n-2} \approx 0.5$, $\lambda_1 \approx 0$.)

The Chebyshev polynomials can be approximated by exponentials: when $j\sqrt{\gamma} > 1$ then, for small γ,

$$C_j(1 + 2\gamma) \approx \tfrac{1}{2}e^{2j\sqrt{\gamma}} \tag{6.6}$$

Using (6.6) in (6.3) and (6.6) yields

$$m - \tfrac{1}{2} = \frac{\ln\tan\psi_n - \tfrac{3}{2}\ln\gamma_n}{2\sqrt{\gamma_n + \gamma_{n-1}}} = \frac{\ln n + 2\ln\omega - \tfrac{3}{2}(\ln\rho + \ln 2)}{4\sqrt{10\rho}} \tag{6.7}$$

where we have written $\tan\psi_n = \omega\tan\psi_{n-1} = \omega\sqrt{n}$. Also

$$k = \frac{\ln n - \ln\rho}{4\sqrt{8\rho}} \approx \frac{\ln n - \ln\rho}{4\sqrt{10\rho}}.$$

These crude approximations give

$$m - k - 1 \approx \frac{2\ln\omega - \tfrac{1}{2}\ln\rho - \tfrac{3}{2}\ln 2}{4\sqrt{10\rho}}. \tag{6.8}$$

In Table 1 we give the predicted and computed values of $m - k - 1$ for interesting values of ρ and ω. As m and k become comparable with n, the use of the Chebyshev polynomial is not warranted, and the predictions become too large, otherwise they are good guides. The point to notice is that misconvergence can occur when $\cos\psi_n$ drops below only 0.1 of its expected value, i.e. $\omega = 10$.

TABLE 1

Predicted and computed values of $m - k - 1$

(*computed values in parentheses*)

$\omega =$	10^1	10^2	10^3
$2^*\rho = 10^{-1}$	2(0)	3(2)	5(5)
$2^*\rho = 10^{-2}$	7(6)	12(12)	17(18)
$2^*\rho = 10^{-3}$	26(17)	42(17)	59(17)
$2^*\rho = 10^{-4}$	95(16)	147(19)	198(17)

The brief discussion is meant to demonstrate how frustrating it is to have to choose an expression for the number of steps to wait before accepting a stagnant interval as containing the largest eigenvalue.

There is a way out which is elegant and also sensitive to each matrix. *Accept the approximation from a stagnant interval as λ_n as soon as a second Ritz value appears in that interval.*

The appearance of superfluous copies of Ritz values is entirely due to roundoff error. A full explanation of this phenomenon is given in [2, Chapter 13] and we do not wish to repeat that material here. Suffice it to say that immediately a Ritz value stagnates at an eigenvalue λ, at step k say, then the algorithm proceeds as though the kth Lanczos vector had a tiny component of λ's eigenvector x. It takes a certain number of extra steps for the component of x in the subsequent Lanczos vectors to grow. When it is dominant another copy of the eigenvalue appears, suddenly, in the form of a new Ritz value hovering at λ.

We know of no cases where a second copy of λ_{n-1} has appeared before some Ritz value exceeds $\lambda_{n-1}(1 + \rho)$. The number of extra Lanczos steps is tolerable in the easy cases but much larger in the difficult cases—as it should be (see Table 5). Nevertheless, this apparent tripling of the number of Lanczos steps was sufficient to deter us, perhaps wrongly. The criterion in Section 2 is slightly more susceptible to misconvergence but keeps the number of steps closer to the minimum.

7. Some Extensions of the Algorithm.

1. The leftmost eigenvalue of T_j, say ω_j, converges to λ_1 in the same fashion as ϑ_j converges to λ_n. The precise rates depend on the eigenvalue distributions. Thus T_j contains increasingly good approximations to cond(A), the condition number of A for linear equations. Several lines of code must be added to compute ω_j at the same time as ϑ_j.

It is wasteful, but feasible, to use our program as it stands on $-A$. We avoided using $\|T_j\|$ in the inner iteration to compute ϑ_j, so that this device would be valid. If the output is negated, it will approximate λ_1.

2. If the α_i and β_i are designated as output parameters and if the Lanczos vectors are put into secondary storage in the course of the Lanczos iteration, then it is easy to compute an approximation to λ_n's eigenvector. The algorithm is:

1. Call (2.4) to compute $\mu \approx \lambda_n$.
2. Call a subroutine to compute the eigenvector s of T belonging to μ, $s = (\sigma_1, \sigma_2, \ldots, \sigma_j)^T$.
3. Set $z = 0$. Recall the Lanczos vectors v_i from storage, one by one, and accumulate $z = z + v_i\sigma_i$.
4. Normalize z.

Since the elements σ_i should dwindle to small values for i close to j, it would be of some slight advantage to recall the Lanczos vectors in reverse order so that the small contributions have a chance to accumulate to significant proportions rather than having their lower order digits lost in the act of adding them to a vector which is already of norm close to one.

3. Our program can be used to find λ_n for the pencil $(K - \lambda M)x = 0$ if M can be factored as LL^T and the matrix A is interpreted as $L^{-1}KL^{-T}$. Of course A is kept in this factored form.

8. Numerical Results. In all our numerical tests** the algorithm described in (2.4) produced the correct answer within the desired accuracy.

In a first series of numerical tests we compared the convergence properties of our algorithm with the power method and Aitken extrapolation applied to the power method. We considered several eigenvalue distributions of matrices of order $n = 500$. Because of the invariance properties of the Lanczos method, there is no loss of generality in considering only diagonal matrices $A = \text{diag}(d_1, d_2, \ldots, d_n)$. We chose $d_i = i$, $d_i = i^2$, $d_i = 1/i$, and $d_i = \cos((i-1)\pi/n)$. In order to give a fair comparison between the different methods, we list in the following table first the number of steps which each method *must* take in order to achieve the correct result within the desired accuracy. The second number, in parentheses, indicates the number of steps taken when the corresponding algorithm is stopped as soon as the increment is $< \frac{1}{2}\rho\lambda_n$. The discrepancy between the two numbers shows how unreliable the old-fashioned stopping criterion can be. The power method does not seem to lend itself to a satisfactory alternative criterion whereas Lanczos does. This is a significant difference.

TABLE 2

Comparison between Lanczos algorithm and power method

(meaning of the numbers is given above the first tableau)

	ρ = rel. accuracy = 10^{-1}			
Eigenvalue distr.	i	i^2	$\frac{1}{i}$	$\cos\left((i-1)\frac{\pi}{n}\right)$
Power	5(4)	5(4)	5(5)	6(4)
Power + Aitken	4(3)	3(4)	3(5)	5(3)
Lanczos	4(5)	4(5)	4(5)	5(5)

	ρ = rel. accuracy = 10^{-3}			
Eigenvalue distr.	i	i^2	$\frac{1}{i}$	$\cos\left((i-1)\frac{\pi}{n}\right)$
Power	95(33)	89(31)	8(9)	490(23)
Power + Aitken	51(22)	44(26)	6(7)	138(19)
Lanczos	23(20)	16(16)	6(6)	48(15)

	ρ = rel. accuracy = 10^{-6}			
Eigenvalue distr.	i	i^2	$\frac{1}{i}$	$\cos\left((i-1)\frac{\pi}{n}\right)$
Power	1169(497)	583(341)	13(14)	>2500(691)
Power + Aitken	467(81)	209(404)	10(11)	>2500(797)
Lanczos	60(48)	43(43)	7(8)	501(117)

From this table we can draw several conclusions. Considering the required accuracy, it seems that it does not pay to use Lanczos if only one digit is wanted. But this conclusion is premature. A quick glance in Table 4 reveals that there are

** All results were obtained at the D.E.C. VAX-11/780 of the Computer Science Division, Electrical Engineering Computer Science Department, University of California, Berkeley.

examples where even for this modest accuracy requirement the power method cannot deliver the correct result. For the sake of reliability, it therefore pays to use the Lanczos algorithm even in these cases.

However, already for three digits accuracy the considerable savings in the number of iteration steps provided by the Lanczos algorithm becomes obvious. This advantage of our algorithm is even more dramatic, if one wants six digits accuracy.

The examples which we have chosen can be regarded as typical for three classes of eigenvalue distributions: easy distributions with the largest eigenvalue well separated from the rest of the spectrum ($d_i = 1/i$), intermediate distributions with all the eigenvalues quite evenly spread out ($d_i = i$, $d_i = i^2$), and difficult distributions with the eigenvalues clustered near the end of the spectrum ($d_i = \cos((i-1)\pi/n)$). Using this classification, one could condense the information from Table 2 in the statement: The more difficult the distribution of the eigenvalues and the higher the desired accuracy is, the more efficient is the Lanczos algorithm.

On the other hand, the numbers in parentheses illustrate quite well that for all three algorithms the most difficult decision is when to stop. Clearly the tolerance $\frac{1}{2}\lambda_n\rho$ is too big.

One might be tempted to use therefore something like $\frac{1}{10}\rho\lambda_n$. But, as pointed out in Section 6, such criteria cannot handle misconvergence properly. In the following table this is illustrated with the example from Section 6. We consider a diagonal matrix of order 100 with $\lambda_n = 1000$, $\lambda_1 = 10$, λ_{n-1} and λ_{n-2} determined from $\gamma_n = 2\rho$, $\gamma_{n-1} = 8\rho$, and the other eigenvalues evenly distributed in $[\lambda_1, \lambda_{n-2}]$. We chose a random starting vector, but set its component in direction of λ_n's eigenvector to ε and normalized it afterwards. For varying values of ρ and ε we obtained the following table. Here the first number indicates the number of steps which were necessary to obtain the correct result, the number in parentheses indicates the number of steps our algorithm stagnated at λ_{n-1}.

TABLE 3

The Lanczos algorithm (2.4) *for the contrived example*

(*stagnation time in paretheses*)

$\varepsilon =$	10^0	10^{-1}	10^{-2}
$2^*\rho = 10^{-1}$	5(0)	8(0)	9(0)
$2^*\rho = 10^{-2}$	13(0)	24(0)	28(6)
$2^*\rho = 10^{-3}$	36(0)	55(4)	59(14)
$2^*\rho = 10^{-4}$	52(0)	68(12)	71(14)

It should be noted that in all the cases in Table 3 our algorithm was capable of handling misconvergence, i.e., the bound $\beta_j\sigma_j$ did not become less than $\frac{1}{2}\rho\lambda_n$ for Ritz values near λ_{n-1}.

The advantages of the Lanczos algorithm together with our stopping criterion will become apparent from the next table.

TABLE 4

Comparison of Lanczos with the power method for the contrived example

Number of Steps	$2\rho =$				
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
Lanczos (correct result)	9	28	59	71	91
Power method (increment $< \rho\lambda_n/2$)	5	3	3	3	4
Aitken extrapol. $(-'-)$	7	6	4	4	4

Here we have contrasted the results of our algorithm from the last column of Table 3 with the corresponding results of the power method and Aitken extrapolation. The latter two methods did not produce the correct results. The increments of these two methods became small as indicated in the table, but both misconverged very soon at λ_{n-1} for at least 150 steps. However, if we chose $\varepsilon = 10^{-3}$ (instead of 10^{-2}), then also our algorithm suffered from misconvergence.

Nevertheless, it is important to stress that this is only due to the stopping criterion we chose. In principle we could have also waited for the appearance of a second copy of a stagnant Ritz value in order to be on the safe side (see Section 6). As this resulted in many cases in a very high number of Lanczos steps, we chose the above error bound stopping criterion as a compromise between reliability and efficiency: It can handle difficult cases in which the power method suffers from misconvergence, and it requires not too many additional Lanczos steps. This is shown in Table 5, where we have listed the number of steps our algorithm took for the matrices from Table 2. In parentheses is the minimum number of steps needed for the given accuracy.

TABLE 5

Results for the Lanczos algorithm (2.4)

(*minimum number of steps needed in parentheses*)

Eigenvalue distr.	i	i^2	$\frac{1}{i}$	$\cos\left((i-1)\frac{\pi}{n}\right)$
$\rho = 10^{-1}$	6(4)	7(4)	5(4)	8(5)
$\rho = 10^{-3}$	46(23)	36(16)	7(6)	140(138)
$\rho = 10^{-6}$	105(60)	76(43)	9(7)	501(501)
Stopping crit. from Section 6 $\rho = 10^{-6}$	143	120	20	>1000

At first glance it might appear that our algorithm (2.4) is taking too many unnecessary steps, but after the preceding discussion of misconvergence it should be clear that this precaution is absolutely necessary. The price one has to pay in order to obtain absolute reliable results is much higher. In the last row we have listed the number of Lanczos steps before a second copy of the dominant Ritz value appeared in a stagnant interval. This waiting time, which seems to guarantee that we do not suffer from misconvergence, costs about 50–100% more Lanczos

steps than our present algorithm. In a sense which deserves to be made more precise we therefore believe that our algorithm is for practical purposes a useful compromise between reliability and efficiency.

Department of Mathematics
Computer Science Division of the
Electrical Engineering Computer Science Department
University of California
Berkeley, California 94720
and
Electrical Research Laboratory
University of California
Berkeley, California 94720

Department of Mathematics
University of California
Berkeley, California 94720

Lawrence Livermore National Laboratories
Livermore, California 94550

1. D. O'LEARY, G. W. STEWART & J. S. VANDERGRAFT, "Estimating the largest eigenvalue of a positive definite matrix," *Math. Comp.*, v. 33, 1979, pp. 1289–1292.

2. B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, N. J., 1980.

3. B. N. PARLETT & J. K. REID, *Tracking the Process of the Lanczos Algorithm for Large Symmetric Eigenproblems, IMA J. Numer. Anal.*, v. 1, 1981, pp. 135–155.

4. L. M. STRINGER, *Efficient and Optimal Methods for Finding the Largest Eigenvalue of a Real Symmetric Matrix*, M.A. Thesis, Univ. of California, Berkeley, 1980.